Landmarks in Planning

Using Landmarks as Pseudo-Heuristics or as Intermediary Goals

Bart van Maris

INTRODUCTION

"What is the performance of using landmarks as intermediary goals and using landmarks as pseudo-heuristics in the SymbolicPlanner framework?"

Based on research done by Richter, Helmert and Wesphal done on Landmarks for Planning algorithms, this research aims to reproduce their work in a new framework to asses the performance of two algorithms mentioned in the work by Richter et al.

PLANNING ALGORITHMS

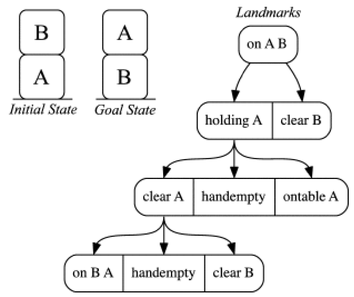
BACKGROUND

Planner:
Make a plan from state A to B in a given domain

Landmark:
Required state to reach B.

Domain:
An environment defined as a set of predicates, axioms and actions.
Used to give context to world. A possible domain can be a map.

Heuristic:
A function that can determine the cost or value of a state.
Examples can be Manhattan distance or Euclidean distance



METHODS

LM Count:
Heuristic counts number of completed landmarks in a state to get heuristic value. The formula goes as follows:

H = N - M + K
- H = Result
- N = Number of landmarks
- M = Completed landmarks
- K = Completed landmarks but needed again.

LM Local:
Uses landmarks as intermediary goals.
Gives goals to internal planner.
Takes the closest sub solution.
Continue from that point.

Implementation:
- Landmarks
- LandmarkNodes
- LandmarkGraphs
- Landmark Extraction
- Landmark Status Manager
- LM Count Heuristic
- LM Local Planner
- LM Local Smart Planner

Performance:
Number of problems solved.
How much time it took to solve

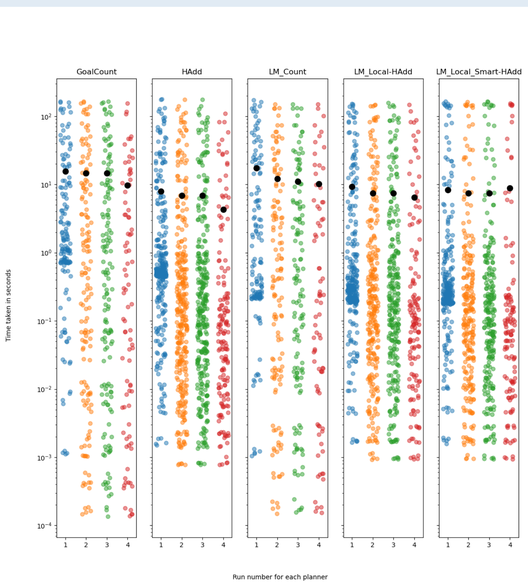
```
function compute(h::LMCount,
                 domain::Domain, state::State, spec::Specification)
    # Progress the Status Manager to update past and Future
    if !(h.prev_state isa State)
        println("Previous state not updated in planner!")
        return 0
    else
        #Previous state of this Heuristic should be updated in the p
        progress(h.lm_status_manager, h.prev_state, state)
        future = get_future_landmarks(h.lm_status_manager, state)
        past = get_past_landmarks(h.lm_status_manager, state)


        h_val = 0
        for i in 1:h.nr_nodes
            if i ∉ past h_val += 1
            elseif i in future h_val += 1 end
        end
        You, 2 weeks ago • Uncommitted changes
        return h_val
    end
end
```

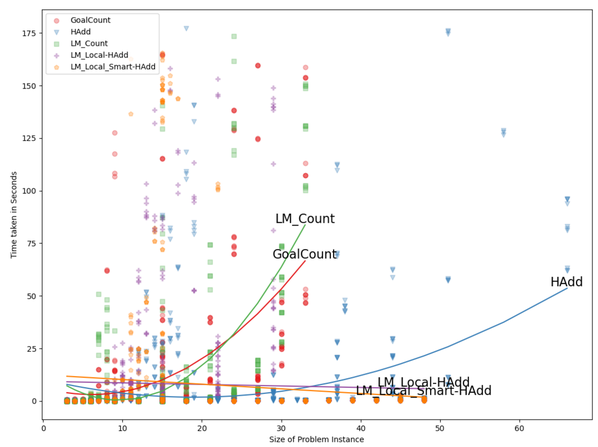
Algorithm 1: LM Local

Data: *planner*
Input: *lm_graph, domain, state, goal_state*
solution
while *lm_graph not empty* **do**
  shortest_sol, used_lm, used_planner
  for *lm* ∈ `get_sources(lm_graph)` **do**
    copy_planner ← *planner*
    sub_sol ←
     copy_planner.`search(domain, state, lm.state)`
    if *sub_sol is shorter than shortest_sol* **then**
      shortest_sol ← *sub_sol*
      used_lm ← *lm*
      used_planner ← *copy_planner*
    end
  end
  remove *used_lm* from *lm_graph*
  planner ← *used_planner*
  solution ← *shortest_sol*
end
solution ←
 planner.`search(domain, state, goal_state)`
return *solution*

RESULTS

- LM Count solves least problem instances, but is faster than GoalCount
- LM Local and LM Local Smart faster than GoalCount and solve more
- LM Local and LM Local Smart keep up with HAdd



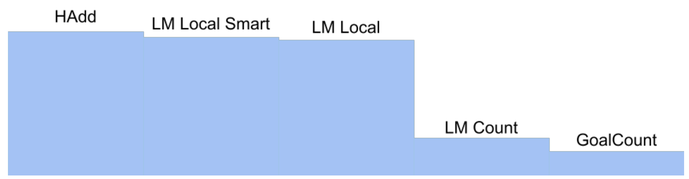


	GoalCount	HAdd	LM Count	LM Local Smart	LM Local
Blocksworld (75)	18	38	15	31	34
Blocksworld_Prodigy (4)	2	4	1	2	4
Freecell (15)	0	0	0	0	0
Grid (5)	0	1	0	0	0
Tyreworld (2)	1	1	1	1	1
Miconic (77)	54	77	53	76	77

CONCLUSION

- HAdd performs best
- LM Local and LM Local Smart are second best
- LM Count only slightly better than GoalCount
- GoalCount clearly worst.

- Our methods could perform better with more landmarks
- LM Count could be combined with other heuristics



ACKNOWLEDGEMENTS

Firstly Paul Tervoort for providing me with his implementation of landmark extraction. He has his own pa-per on this specific topic. Secondly the rest of our re-search group, Pauline Hengst, Noah Tjoen and Ka Fui Yang,for support and insightful discussions on our shared topic of landmarks. Thirdly Issa Hanou and Sebastijan Dumancic, our supervisor and responsible professor for providing feedback and answering any questions we had during the course of this project.